Localisation précise et temps réel dans un environnement partiellement connu : application au suivi d'objet 3D peu texturé

Mohamed Tamaazousti, Vincent Gay-Bellile, Naudet Collette Sylvie, Steve Bourgeois, Michel Dhome

Localisation précise et temps réel dans un environnement partiellement connu : application au suivi d'objet 3D peu texturé

Mohamed Tamaazousti[1] Vincent Gay-Bellile[1] Sylvie Naudet Collette[1] Steve Bourgeois[1]
Michel Dhome[2]

[1]CEA, LIST, Laboratoire Vision et Ingénierie des Contenus
Point Courrier 94, Gif-sur-Yvette, F-91191 France
[2]Clermont Université, Université Blaise Pascal, LASMEA, BP 10448
Clermont-Ferrand / CNRS, UMR 6602, LASMEA, AUBIERE

mohamed.tamaazousti@cea.fr

Résumé

Ce papier a pour sujet la localisation temps réel d'une caméra dans un environnement partiellement connu, c'est-à-dire pour lequel un modèle géométrique 3D d'un objet statique de la scène est disponible. Nous proposons de tirer avantage de ce modèle géométrique pour améliorer la précision de la localisation par un algorithme de SLAM basé images clefs en incluant dans le processus d'ajustement de faisceaux cette information additionnelle. Afin de pouvoir gérer des objets 3D peu texturés, une contrainte avec des segments 3D extraits du modèle est proposée ici. Les avantages de cet ajustement de faisceaux contraint par segments sont démontrés sur des données de synthèse et réelles. Des applications, temps réel, de réalité augmentée sont également présentées sur des objets 3D peu texturés.

Mots Clef

SLAM, modèle 3D, ajustement de faisceaux local, temps réel.

Abstract

This paper addresses the challenging issue of real-time camera localization in a partially known environment, i.e. for which a geometric 3D model of one static object in the scene is available. We propose to take benefit from this geometric model to improve the localization of keyframe-based SLAM by constraining the local bundle adjustment process with this additional information. The constraint considered here deals with textureless 3D objects. We demonstrate the advantages of our constrained bundle adjustment framework on both synthetic and real data and present very convincing augmentation of textureless 3D objects in real-time.

Keywords

SLAM, 3D model, Local Bundle Adjustment, Real Time.

1 Introduction

Le suivi d'objet 3D avec une caméra en mouvement est un sujet de recherche très actif. Il requiert d'estimer les poses relatives de la caméra par rapport à cet objet 3D et ceci en temps réel. Les solutions basées modèle exploitent une connaissance *a priori* de la géométrie et/ou de l'apparence de l'objet pour estimer cette pose en mettant en correspondance des primitives 3D du modèle avec celles extraites dans l'image courante [7]. Néanmoins, pour un suivi stable, ce processus implique que l'objet d'intérêt soit visible de manière continue et tienne une place prépondérante dans les images durant l'ensemble de la séquence.

D'un autre côté, les algorithmes de SLAM basé images clefs, examples [5, 10, 11], estiment le mouvement relatif de la caméra sans information *a priori* sur la géométrie de la scène. Ils exploitent les relations multi-vues pour estimer itérativement le mouvement de la caméra et l'environnement (sous forme d'un nuage de points 3D éparses), ces données sont ensuite raffinés avec un ajustement de faisceaux (BA pour *Bundle Adjustment*). Alors que les solutions hors ligne [11] utilisent un ajustement de faisceaux global qui raffine l'ensemble de la reconstruction de la séquence vidéo dans un unique processus d'optimisation, des solutions en ligne [5, 10] utilisent un ajustement de faisceaux local qui optimise de manière séquentielle un nombre limité de poses de caméra et les points 3D qu'elles observent. La localisation avec un SLAM basé images clefs est très stable vu que l'ensemble de l'information présente dans les images est utilisée pour estimer le déplacement de la caméra. Cependant, les solutions de SLAM basé images clefs sont sujettes à trois limitations majeures qui les rendent inutilisable dans un contexte de suivi d'objet 3D :

– le repère initial est choisi de manière arbitraire
– le facteur d'échelle initial de la scène est arbitrairement fixé
– la localisation est sujette à une accumulation d'erreur dû

au bruit dans les images, aux erreurs d'appariement, *etc*.

Récemment, différents travaux ont tenté de résoudre certaines de ces limitations en combinant SLAM et suivi basé modèle. Bleser et al. [1] ont introduit une solution pour fixer le repère initial et l'échelle de la reconstruction en utilisant une méthode de suivi basée modèle sur la première image. Une fois que la pose initiale de la caméra est estimée (dans le repère de l'objet), le facteur d'échelle est alors transmis au SLAM par rétro projection des points d'intérêt extraits de la première image, sur la surface du modèle. Il en résulte, un nuage de points 3D qui est alors utilisé comme reconstruction initiale de l'environnement par l'algorithme de SLAM. Néanmoins, cette solution présente différentes limitations : premièrement, le repère initial ainsi que le facteur d'échelle sont estimés à partir d'un unique point de vue qui peut être sujet à des imprécisions, tout particulièrement le long de l'axe optique de la caméra. Deuxièmement, le problème d'accumulation d'erreur n'est pas traité.

Pour traiter ce dernier problème, il a récemment été montré, dans [12], qu'il est possible de bénéficier d'un modèle géométrique pour améliorer la localisation d'un algorithme de SLAM basé images clefs. Un processus d'ajustement de faisceaux contraint est alors introduit. Il inclut simultanément les contraintes géométriques fournies par le modèle 3D, les relations multi-vues relatives à la partie connue de l'environnement (c'est-à-dire les observations de l'objet d'intérêt dans la séquence vidéo) et les relations multi-vues relatives à la partie inconnue de l'environnement (c'est-à-dire les autres observations). De manière plus spécifique, la contrainte proposée dans [12] impose à chaque point 3D associé à l'objet d'intérêt, d'appartenir à un des plans du modèle 3D. Cette contrainte est directement intégrée dans le processus d'ajustement de faisceaux. Cette approche résout les limitations des méthodes de type SLAM, dans le contexte de suivi d'objet 3D. Cependant, l'algorithme proposé dans [12] présente une limitation majeure : l'objet doit être suffisamment texturé. En effet, les contraintes planaires nécessitent qu'un nuage de points 3D de l'objet d'intérêt ait été précédemment reconstruit par le SLAM.

Dans ce papier nous introduisons un nouvel ajustement de faisceaux contraint qui permet de gérer des objets 3D peu texturés. Dans ce cas, une contrainte basée modèle fournie par les contours francs du modèle 3D est utilisée. Des segments extraits du modèle 3D sont utilisés dans l'ajustement de faisceaux pour contraindre la trajectoire de la caméra en ajoutant leurs erreurs de reprojection dans la fonction de coût. La structure creuse de l'ajustement de faisceaux local contraint, ainsi formulé, est exploitée dans l'implémentation afin d'assurer des performances temps réel. La solution ici présentée, résout le problème du recalage du repère initial, de l'estimation du facteur d'échelle, de l'accumulation d'erreur et des larges occultations. Elle est donc parfaitement adaptée pour des applications de réalité augmentée sur des objets 3D peu texturés.

Plan. La Section 2, décrit brièvement les algorithmes de SLAM basé images clefs et les processus d'ajustement de faisceaux local/global associé. La Section 3 présente le processus d'ajustement de faisceaux contraint par segments. La Section 4, détaille les étapes additionnelles requises par le processus d'ajustement de faisceaux contraint. La Section 5 étudie la structure creuse de l'ajustement de faisceaux contraint par segments. Le processus d'ajustement de faisceaux contraint est évalué, sur des données de synthèse ainsi que sur des données réelles, pour le suivi d'objets 3D peu texturés, dans la Section 6. Finalement, nous concluons et discutons des travaux futurs, dans la Section 7.

Notation. Les matrices sont représentées par des caractères sans empattements comme M. Les vecteurs sont représentés en gras et exprimés en coordonnées homogènes, exemple $\mathbf{q} \sim (x,y,w)^\mathsf{T}$ où $^\mathsf{T}$ est la transposée et $\sim$ l'égalité à un facteur d'échelle près. La reconstruction SLAM est composée de N points 3D $\{\mathbf{Q}_i\}_{i=1}^N$ et de m caméras $\{\mathsf{C}_k\}_{k=1}^m$. Nous notons $\mathbf{q}_{i,k}$ l'observation du point 3D $\mathbf{Q}_i$ dans la caméra C_k et $\mathscr{A}_i$ l'ensemble des indices de caméra observant $\mathbf{Q}_i$ avec $n_i = \text{card}(\mathscr{A}_i)$. La matrice de projection P_k associée à la caméra C_k est donnée par $\mathsf{P}_k = \mathsf{K}\mathsf{R}_k^\mathsf{T}(\mathsf{I}_3|-\mathbf{t}_k)$, où K est la matrice des paramètres intrinsèques et $(\mathsf{R}_k,\mathbf{t}_k)$ les paramètres extrinsèques.

2 SLAM basé images clefs

Les algorithmes de SLAM basé images clefs, examples [5, 10, 11], sont particulièrement appropriés pour la localisation d'une caméra dans un environnement inconnu. Ils permettent d'estimer simultanément la trajectoire de la caméra ainsi que la géométrie de la scène observée (sous forme d'un nuage de points 3D). Pour y parvenir, ces algorithmes se déroulent généralement en plusieurs étapes : l'initialisation, le suivi 3D, la triangulation et l'optimisation. L'étape d'initialisation permet d'obtenir une première reconstruction avec deux ou trois images. Cette reconstruction fixe de manière arbitraire le repère global et l'échelle de la reconstruction. Après cela, une estimation de pose robuste est réalisée pour chaque image de la vidéo en utilisant une détection et un appariement des primitives. La triangulation [4] des points 3D s'effectue uniquement aux images dites clefs. Ces images sont choisies de manière à ce qu'il y ait suffisamment d'appariements de points 2D pour assurer le calcul de pose de la caméra et un déplacement inter caméra suffisamment important pour une triangulation précise. Ensuite, les poses de la caméra et le nuage de points 3D sont simultanément raffinés par un ajustement de faisceaux. Ce dernier minimise la somme des différences au carré entre la projection des points 3D et leurs observations dans les images. Cette distance géométrique est appelée erreur de reprojection. Les paramètres d'optimisation sont les N points 3D et les 6 paramètres extrinsèques des m poses de la caméra. Ainsi le nombre total de paramètres est de $3N+6m$. La fonction de coût de l'ajustement de faisceaux est donnée par :

$$\mathscr{E}_E\left(\{\mathsf{R}_j,\mathbf{t}_j\}_{j=1}^{m},\{\mathbf{Q}_i\}_{i=1}^{N}\right)=\sum_{i=1}^{N}\sum_{j\in\mathscr{A}_i}d^2(\mathbf{q}_{i,j},\mathsf{P}_j\mathbf{Q}_i),\quad(1)$$

où $d^2(\mathbf{q},\mathbf{q}')=\|\mathbf{q}-\mathbf{q}'\|^2$ est la distance point-point.
Pour les séquences de grande taille, BA devient rapidement très couteux en temps de traitement et n'est donc pas adapté à une localisation temps réel et cela même avec la prise en compte de la structure creuse lors de l'implémentation (pour les détails, voir plus bas). Pour palier cette limitation, un processus d'ajustement de faisceaux local a été proposé par [10]. L'idée est de réduire le nombre des paramètres à estimer en optimisant uniquement un sous ensemble des points 3D reconstruits et des poses de la caméra. Dans son implémentation originale, les paramètres à optimiser sont les T plus récentes poses de la caméra (où T est choisi de manière à maintenir des performances de traitement temps réel [1]) et les points 3D qu'elles observent.

Equations normales et structure creuse. Eq. 1 est minimisée de manière itérative. La fonction est linéarisée à chaque itération au voisinage des paramètres courants en effectuant une décomposition de Taylor à l'ordre deux ; dans la méthode de Gauss-Newton la matrice des dérivées seconde, c'est-à-dire la matrice Hessienne, est approximée par le produit des Jacobiens. Le système résolu à chaque itération est alors le suivant :

$$\begin{pmatrix}\mathsf{U} & \mathsf{W}\\ \mathsf{W}^\mathsf{T} & \mathsf{V}\end{pmatrix}\begin{pmatrix}\Delta_c\\ \Delta_p\end{pmatrix}=\begin{pmatrix}\mathsf{J}_c^\mathsf{T}\varepsilon\\ \mathsf{J}_p^\mathsf{T}\varepsilon\end{pmatrix}\quad(2)$$

où $\mathsf{U}=\mathsf{J}_c{}^\mathsf{T}\mathsf{J}_c$, $\mathsf{V}=\mathsf{J}_p{}^\mathsf{T}\mathsf{J}_p$, $\mathsf{W}=\mathsf{J}_c{}^\mathsf{T}\mathsf{J}_p$ sont les sous matrices qui composent l'approximation de la matrice Hessienne, $\mathsf{J}=[\mathsf{J}_c,\mathsf{J}_p]$ est la matrice Jacobienne, Δ_c, Δ_p sont les incréments des paramètres à estimer et ε est un vecteur concaténant toutes les erreurs résiduelles [2].
Nous utilisons, l'algorithme de Levenberg-Marquardt (LM) [9] pour lequel il faut résoudre une équation très peu différente de Eq. 2. Chaque itération combine la méthode de Gauss-Newton et la méthode de descente de gradient. En effet, dans le cas de l'algorithme LM les termes diagonaux de la matrice $\mathsf{J}^\mathsf{T}\mathsf{J}$ sont multipliés par $(1+\alpha)$, où $\alpha\in\mathbb{R}$ et $\alpha>0$. Le fait de modifier ainsi la diagonale de la matrice est appelé "damping", et le paramètre α est dit "coefficient d'amortissement". Augmenter (resp. réduire) α revient à donner plus (resp. moins) d'importance à la descente de gradient. Lourakis propose dans [8], une heuristique pour initialiser et mettre à jour ce paramètre.
Une propriété utile de l'ajustement de faisceaux est la structure par blocs des matrices mises en jeux, dû au fait que chaque résidu implique uniquement un point 3D et une pose de caméra. Ainsi, les matrices U et V sont diagonales par blocs avec respectivement des blocs de tailles 6×6 et 3×3. La matrice W est composée de sous matrices 6×3 qui expriment les inter corrélations entre la structure et les paramètres de déplacement de la caméra. En tenant compte de la spécificité de cette structure par blocs, BA peut être implémenté de manière efficace comme décrit dans [13].
La Figure 1 représente la structure par blocs des matrices Jacobienne et Hessienne sur un exemple d'illustration.

1. $T=3$ dans [10].
2. où les indices $_c$ et $_p$ représentent respectivement la dépendances aux paramètres des poses de la caméra et des points 3D.

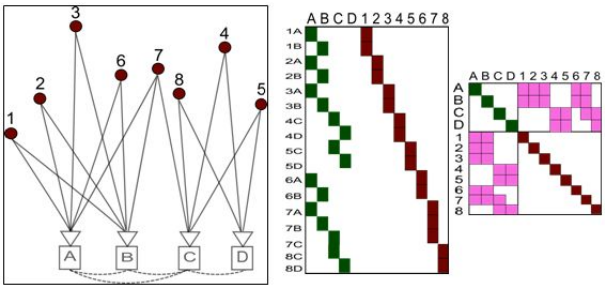

FIGURE 1 – A gauche : un exemple d'ajustement de faisceaux. En rouge foncé, le nuage de points 3D reconstruit $(1-8)$. Les carrés (A-D) représentent la trajectoire de la camera. Les lignes noires indiquent l'observabilité d'un point 3D à partir d'une certaine pose de la caméra en mouvement. A droite : les matrices Jacobienne et Hessienne associées.

3 Ajustement de faisceaux contraint par segments

Formulation du problème. Les limites majeures, pour le suivi d'objet, des algorithmes de type SLAM classique tels que présentés dans la Section 2, sont que la reconstruction est estimée à un facteur d'échelle prés et que le repère initial est fixé de manière arbitraire. Ils sont également sujets à une accumulation d'erreurs dues au bruit dans les images et aux erreurs d'appariement, *etc*. Pour palier à ces limitations, il est possible de bénéficier des contraintes géométrique fournies par le modèle 3D d'un objet statique de la scène observée, dans un ajustement de faisceaux contraint. Il en résulte une fonction de coût composée de la partie connue (c'est-à-dire avec des contraintes basé modèle) et de la partie inconnue (c'est-à-dire sans contrainte basé modèle) de l'environnement :

$$\mathscr{E}=\mathscr{E}_E+\lambda\mathscr{E}_M,\quad(3)$$

où $\mathscr{E}_E$,$\mathscr{E}_M$ sont respectivement les termes associées aux parties connue et inconnue de l'environnement, et λ est le poids qui contrôle l'influence du terme associé au modèle.
Le SLAM avec l'ajustement de faisceaux contraint possède plusieurs avantages. En effet, non seulement il résout les limitations, des méthodes de SLAM "classique", mentionnées précédemment, mais il permet également de corriger des erreurs générées lors du recalage initial des repères.

Choix de la contrainte. La contrainte présentée ici utilise des segments extraits du modèle 3D pour contraindre directement les poses de la caméra. La contrainte consiste à

aligner les contours francs du modèle 3D avec les contours de l'objet dans l'image. Les contours francs du modèle 3D (c'est-à-dire pour un modèle en facette ou *mesh*) sont les contours formés par deux triangles et dont l'angle diédral est supérieur à un certain seuil. Ces contours sont ensuite échantillonnés en un ensemble de petits segments 3D $\{\mathbf{L}_i\}_{i=1}^{s}$; chacun étant paramétrisé par un point milieu $\mathbf{M}_i$ et une direction $\mathbf{D}_i$. Ces segments constituent, la partie connue de l'environnement. Ainsi, cette contrainte permet de gérer le suivi d'objets 3D peu texturés.

Fonction de coût. La fonction de coût minimise la distance orthogonale entre le point milieu du segment projeté $\mathsf{P}_j\mathbf{M}_i$ et le contour $\mathbf{m}_{i,j}$ auquel il est associé dans l'image, voir [15] pour plus de détails. Considérons des associations 2D/3D entre des segments 3D $\mathbf{L}_i$ du modèle et des contours $\mathbf{m}_{i,j}$ extraits aux images clefs $j \in \mathscr{S}_i$ (voir la section 4.1), la contrainte par des segments est donnée par :

$$\mathscr{E}_M\left(\left\{\mathsf{R}_j,\mathbf{t}_j\right\}_{j=1}^{m}\right) = \sum_{i=1}^{s}\sum_{j\in\mathscr{S}_i}\left|\mathbf{n}_{i,j}.(\mathbf{m}_{i,j}-\mathsf{P}_j\mathbf{M}_i)\right|, \quad (4)$$

où $\mathbf{n}_{i,j}$ est la normale de la projection de la direction $\mathsf{P}_j\mathbf{D}_i$. La fonction de coût est exprimée en pixel pour faciliter la pondération des différents termes de la fonction de coût totale (voir Section 4.2). Comparativement aux méthodes de suivi basées contours du modèle [2, 3, 15], cette fonction de coût utilise sensiblement le même critère mais étendu au cas multi-vues : il s'applique simultanément sur plusieurs vues.

A noter que, les segments ont précédemment été utilisés par Klein *et al.* dans [6] pour améliorer la robustesse du SLAM aux mouvements de caméra rapide. Ils proposent alors un processus complet, incluant extraction de contour et appariement dans les images clefs successives, triangulation des segments et un ajustement de faisceaux dédié à combiner les points et les segments. Dans notre cas, nous possédons déjà une carte de segments 3D fournis par le modèle géométrique de l'objet d'intérêt. Nous utilisons cette information disponible pour améliorer la précision des algorithmes SLAM basé images clefs.

4 Etapes additionnelles de l'ajustement de faisceaux contraint par segments

L'ajustement de faisceaux contraint ainsi proposé requiert des étapes supplémentaires comparé à l'ajustement de faisceaux "classique". En effet, pour introduire la contrainte basée modèle, les primitives doivent être associées au modèle. De plus, comme l'illustre l'Eq. (6) nous sommes dans le cas d'une résolution d'un problème bi-objectif. Il requiert la gestion de l'influence de chacun des deux termes $\mathscr{E}_E$ et $\mathscr{E}_M$ durant sa minimisation. Cette section, décrit les étapes additionnelles nécessaires pour le processus d'ajustement de faisceaux contraint par segments. Celles ci sont : un processus d'associations de données au modèle, une estimation robuste pour gérer les mauvaises associations et un schéma de pondération des deux termes de la fonction de coût global.

4.1 Associations 2D/3D entre les contours image et les segments 3D

Cette étape d'association 2D/3D est réalisée uniquement aux images clefs pour le processus d'ajustement de faisceaux contraint. Pour l'Eq. 4, les segments 3D extraits du modèle doivent être associés aux contours dans l'image. Un test de visibilité est utilisé dans un premier temps pour ne traiter qu'un sous ensemble de segments 3D visibles pour chaque image clefs du BA. Les segments 3D visibles sont alors projetés dans chaque image clef, puis une recherche est réalisée, le long de la normale au segment projeté, afin de trouver le maximum de gradient, dans un petit voisinage. En pratique nous observons que choisir uniquement le contour le plus proche avec une orientation compatible avec celle prédite (c'est à dire en deçà d'un seuil fixé) est suffisant, puisque la pose initiale estimée par le SLAM est proche de la solution. En effet, à la différence des méthodes de suivi basées modèle, la pose initiale n'est pas celle de l'image précédente, mais elle a été calculée à partir d'appariements 2D/3D. Cette pose initiale est donc plus précise, ce qui évite les ambigüités de mise en correspondance liées à une pose trop approximative.

4.2 Estimation robuste et pondération

Un mauvais recalage initial (entre le repère monde et le repère de l'objet) et la dérive inhérente aux algorithmes de type SLAM peuvent introduire de mauvaises associations 2D/3D, perturbant la convergence du processus d'optimisation. Pour rendre la méthode robuste à des associations aberrantes, une estimation robuste est effectuée avec le M-estimateur de Geman-McClure $\rho(r,c) : \mathbb{R} \rightarrow [0 \cdot\cdot 1]$ où

$$\rho(r,c) = \frac{r^2}{r^2+c^2}, \quad (5)$$

avec, r est l'erreur résiduelle d'une des fonctions de coût décrites précédemment (1) ou (4), et c le seuil de rejet. Il est estimé automatiquement en utilisant la médiane des valeurs absolue (Median Absolute Deviation MAD). Notons que le MAD suppose une distribution normale des résidus. Combiner l'Eq. (1) avec l'Eq. (4) n'est pas évident même si ces différentes fonctions de coût sont exprimées en pixel. En effet, elles n'ont pas forcement le même ordre de grandeur : les erreurs résiduelles associées à la partie connue de l'environnement sont généralement plus élevées.

La pondération des deux termes est calculée comme dans [12], qui propose une solution originale basée sur un choix particulier du seuil de rejet de l'estimateur robuste. Ce seuil est réestimé à chaque ajustement de faisceaux local.

Les étapes de l'ajustement de faisceaux contraint par segments sont récapitulées dans le Tableau 1. La fonction de coût minimisée est décrite par Eq. (6).

$$\mathscr{E}\left(\left\{\mathsf{R}_j,\mathbf{t}_j\right\}_{j=1}^{m},\left\{\mathbf{Q}_i\right\}_{i=1}^{N}\right)=\underbrace{\sum_{i=1}^{N}\sum_{j\in\mathscr{A}_i}\rho\left(d^2(\mathbf{q}_{i,j},\mathsf{P}_j\mathbf{Q}_i),c_1\right)}_{\text{Partie inconnue de l'environnement (E)}}+\underbrace{\sum_{i=1}^{s}\sum_{j\in\mathscr{S}_i}\rho\left(\left|\mathbf{n}_{i,j}.\left(\mathbf{m}_{i,j}-\mathsf{P}_j\mathbf{M}_i\right)\right|,c_2\right)}_{\text{Partie connue de l'environnement (CS)}} \quad (6)$$

– Test de visibilité pour trouver le sous ensemble de segments 3D visible pour chaque image clef $j \in \mathscr{S}_i$.
– Projection des segments 3D visible $\mathbf{L}_i$ dans chaque image clef $j \in \mathscr{S}_i$.
– Association des segments 3D $\mathbf{L}_i$ aux contours de l'image $\mathbf{m}_{i,j}$.
– Calcul des seuils de rejet c_1 et c_2.
– Minimisation des Eq. (6) avec l'algorithme de Levenberg Marquardt (LM)[9].

TABLE 1 – Ajustement de faisceaux contraint par segments.

5 Structure creuse de l'ajustement de faisceaux contraint par segments

Cette section, montre que certaines propriétés des matrices Hessienne et Jacobienne, associées à l'ajustement de faisceaux, sont conservées malgré l'ajout de la contrainte par segments.

La Figure 2 représente un exemple d'illustration d'ajustement de faisceaux contraint par segments ainsi que les matrices Hessienne et Jacobienne mises en jeux. Une propriété importante est que ces matrices conservent une structure par blocs. En effet, des résidus dépendant uniquement des paramètres de la caméra sont ajoutés dans la matrice Jacobienne (un résidu pour chaque observation d'un segment 3D). De plus, le nombre de paramètres n'augmente pas puisque les segments 3D ne sont pas optimisés. La conséquence est que la matrice Hessienne possède exactement la même structure que pour un ajustement de faisceaux classique.

Ainsi, il est possible d'implémenter de manière efficace l'ajustement de faisceaux contraint, en prenant en compte cette structure par blocs, comme décrit dans [13].

6 Résultats expérimentaux sur des données de synthèse et réelles

Dans cette section, des évaluations de l'ajustement de faisceaux contraint sont présentées sur des données de synthèse et réelles. L'algorithme de SLAM basé images clefs, tel que décrit dans [10], est utilisé dans les différentes expérimentations. Il fonctionne en temps réel grâce à un ajustement de faisceaux local appliqué sur une fenêtre glissante de triplet d'images clefs. A chaque image clef, seules les trois dernières poses de la caméra (associées aux trois dernières images clefs) ainsi que les points 3D qu'elles observent, sont optimisés.

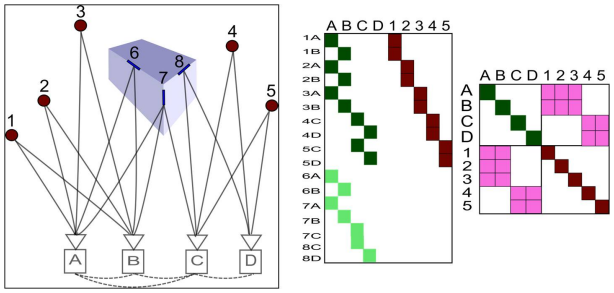

FIGURE 2 – A gauche : un exemple d'ajustement de faisceaux contraint par segments. En rouge foncé, les points 3D de la partie inconnue de l'environnement $(1-5)$; en bleu foncé, les segments 3D associés à la partie connue de l'environnement $(6-8)$. Les carrés (A-D) représentent la trajectoire de la camera. Les lignes noires indiquent l'observabilité d'une primitive 3D (point ou segment) à partir d'une certaine pose de la caméra en mouvement. A droite : les matrices Jacobienne et Hessienne associées.

Une évaluation des performances, pour le suivi d'objet 3D, de l'algorithme de SLAM, est réalisée sur une séquence de synthèse, avec deux types d'ajustement de faisceaux local :

– Celui décrit dans [10] dans sa version originale appelé par la suite LBA_E. Il minimise Eq. (1) par l'algorithme de Levenberg-Marquardt.
– L'algorithme LBA_CS&E[3] proposé ici. Il minimise Eq.(6) avec la procédure décrite dans le Tableau 1.

Une comparaison est ensuite effectuée, sur des données réelles entre l'algorithme de SLAM avec l'ajustement de faisceaux contraint (SLAM + LBA_CS&E) et l'état de l'art en suivi d'objet 3D.

A noter que, l'étape d'initialisation de [10], a été modifiée. Dans la version initiale, elle est réalisée par l'algorithme des cinq points, sur les trois premières images clefs. Elle est ici réalisée sur la première image où une estimation approximative de la première pose de la caméra est calculée. Cela permet ainsi de recaler le repère global avec celui du modèle 3D de l'objet. Un nuage de points 3D initial est ensuite obtenu en faisant une rétro projection des observations de la première image.

6.1 Evaluation sur des données de synthèse

Cette section détaille l'expérimentation réalisée pour comparer les deux algorithmes d'ajustement de faisceaux

3. CS signifie Contrainte par des Segments pour les contraintes au modèle, c'est-à-dire la partie connue de l'environnement et E signifie que les relations multi-vues de la partie inconnue de l'environnement sont prises en compte.

LBA_E et LBA_CS&E, sur une séquence de synthèse. Après avoir décrit brièvement la séquence d'étude, l'apport de l'algorithme LBA_CS&E dans un SLAM est évalué, en terme de précision de la localisation et en terme de robustesse à une initialisation approximative. La qualité de la localisation est mesurée par le RMS 3D entre la vérité terrain et la pose estimée après l'ajustement de faisceaux local à chaque image clef.

La séquence "cubes". Dans cette séquence, illustrée par la Figure 3, la scène est composée de deux cubes situés sur un sol texturé avec d'autres petits cubes qui les occultent partiellement. L'objet d'intérêt, c'est-à-dire pour lequel un modèle 3D est disponible, est composé des deux cubes principaux. La trajectoire de la caméra est un cercle dont le rayon est de trois mètres autour des principaux cubes. La hauteur des deux cubes superposés est d'un mètre et demi.

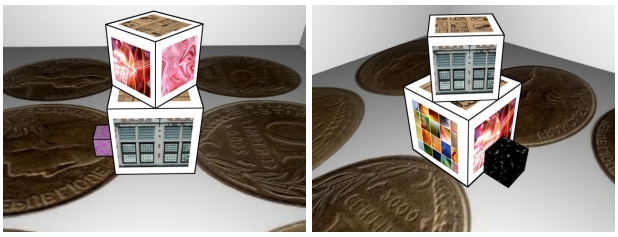

FIGURE 3 – Deux images de la séquence "cubes".

Comparaison des deux algorithmes LBA. Deux types d'expérimentations ont été réalisées sur la séquence "cubes" pour comparer les algorithmes de raffinement LBA_E et LBA_CS&E. La première évalue la précision de la localisation au cours de la séquence avec une initialisation parfaite donnée par la vérité terrain. Les résultats sont présentés sur les Figures 4 (a) (position) et 4 (b) (orientation). Le SLAM avec l'algorithme de raffinement LBA_E est sujet à des accumulations d'erreurs alors qu'avec l'ajustement de faisceaux local contraint LBA_CS&E, il ne dérive pas. L'ajout de la contrainte améliore de manière significative la localisation.
La seconde expérimentation, évalue la robustesse du SLAM + LBA_CS&E à une initialisation approximative. Des perturbations de plus en plus élevées sont appliquées à la pose initiale de la caméra, correspondant à la première image. Leurs amplitudes varient de 1% à 6% du rayon du cercle formé par la trajectoire de la caméra (un exemple de perturbation de 6% est illustré sur la Figure 4 (d)). Les résultats sont une moyenne sur dix tirages dans des directions aléatoires. La Figure 4 (c) montre que le SLAM avec l'ajustement de faisceaux local contraint par segments gère des erreurs d'initialisation : après quelques poses de la caméra les erreurs 3D sont stabilisées à de faibles valeurs. Notons que le résultat concernant la robustesse à l'initialisation pour l'algorithme LBA_E, n'est pas présenté ici car il est évident qu'il ne peut pas corriger cette erreur d'initialisation.

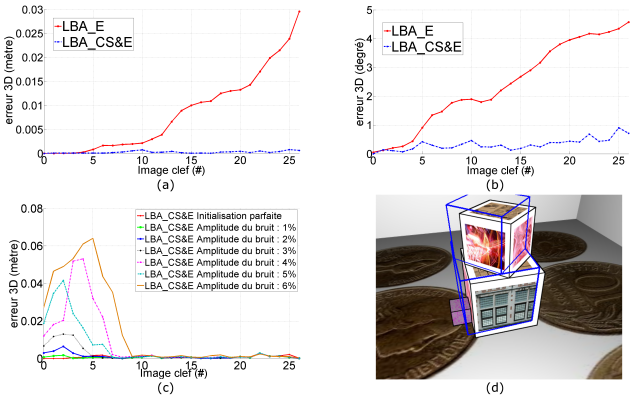

FIGURE 4 – (a) (resp. (b)) : erreurs en position (resp. orientation) exprimées en mètre (resp. en degré) pour le SLAM avec les deux algorithmes de LBA. (c) les résultats obtenus par le SLAM avec l'algorithme de raffinement LBA_CS&E pour différentes amplitudes de perturbation ($\in [1\% \cdots 6\%]$ du rayon de la trajectoire de la caméra) appliquées sur la première pose de la caméra. (d) est une illustration du décalage dans l'image entre la reprojection du modèle et l'objet d'intérêt, pour une perturbation de 6%.

6.2 Evaluation sur des données réelles

Il a été montré, ci dessus, qu'avec l'algorithme de raffinement LBA_CS&E, les approches de type SLAM peuvent être utilisées pour le suivi d'objet 3D, dans un environnement partiellement connu. Dans cette section, le SLAM + LBA_CS&E est alors comparé à l'état de l'art pour les méthodes de suivi basées modèle, pour le suivi d'objet 3D peu texturé. Pour cela, des séquences ont été réalisées avec une caméra IEEE1394 GUPPY, bas coût, fournissant des images (640×480) à une fréquence de 30 images par seconde.

Suivi d'objet 3D peu texturé. L'objet d'intérêt est une maquette miniature représentant la Lamborghini Gallardo. Le modèle 3D utilisé dans les expérimentations est composé d'environ 14000 triangles. 1186 segments 3D, sont alors extraits de celui ci, comme illustré sur la Figure 5. La voiture est placée sur un bureau composé d'un écran et clavier d'ordinateur, de livres, *etc*. Ils constituent la partie inconnue de l'environnement. Une comparaison est effectuée entre le SLAM avec LBA_CS&E et un algorithme de suivi basé modèle, qui par définition utilise uniquement la partie connu de l'environnement. Ce dernier est une version améliorée de l'algorithme de suivi basé modèle proposé par [2] qui inclut, en plus, une notion d'hypothèses multiples pour l'associations de données, comme décrit dans [14, 16]. La comparaison est réalisée sur une séquence difficile qui présente des variations d'échelle importantes, des mouvements rapides, des occultations partielles de la voiture, *etc*. A noter que, comme l'objet d'intérêt est peu texturé, l'initialisation (c'est-à-dire le recalage sur la première image)

est obtenu en positionnant approximativement, pour la première image, un marqueur codé prés de la voiture.

FIGURE 5 – Les segments 3D extraits du modèle de la Lamborgini.

Résultats. La Figure 6 présente les résultats obtenus par le SLAM avec LBA_CS&E et par l'algorithme de suivi basé modèle, sur cette séquence. Le marqueur codé étant positionné approximativement, prés de la voiture, le recalage initial sur la première image n'est donc pas précis. Les deux méthodes corrigent cette erreur après quelques images : l'avant et l'arrière de la voiture se projettent correctement sur les images, comme présenté sur la Figure 6 (à gauche).
Cependant, le SLAM avec LBA_CS&E fournit de meilleurs résultats que l'algorithme de suivi basé modèle. En effet, ce dernier est mis en échec lors de mouvements rapides de la caméra (voir la Figure 6 à droite). Ceci est dû à de mauvaises mises en correspondances 3D/2D. De plus, la localisation obtenue est instable, ce qui se traduit par des tremblements du modèle reprojeté dans les images. L'algorithme de localisation que nous proposons utilise simultanément les informations des parties connue et inconnue de l'environnement. Il en résulte une localisation précise, stable et robuste. L'algorithme SLAM avec LBA_CS&E est donc parfaitement adapté pour des applications de réalité augmentée en temps réel.

6.3 Application à la réalité augmentée

Nous présentons deux applications de réalité augmentée par la méthode SLAM, avec l'algorithme LBA_CS&E proposé. Une première est la personnalisation virtuelle d'un véhicule comme illustré sur la Figure 7. La personnalisation de la voiture est effectuée en changeant la couleur d'un ou de plusieurs éléments de la carrosserie. Il est ainsi possible de changer séparément la couleur du capot, du coffre et de toute la carrosserie de la voiture, de manière très réaliste. Notons que pour une meilleure appréciation de la précision de la méthode, les roues, les fenêtres et le pare-brise ne sont pas augmentés.
L'algorithme de localisation proposé est également appliqué au design de cuisine par réalité augmentée. Le but est de permettre à l'utilisateur de voir les différentes combinaisons possible d'une cuisine c'est-à-dire couleur, matériaux,

FIGURE 6 – Localisation dans un environnement partiellement connu, composé d'un objet 3D non texturé. En haut : les résultats obtenues avec un suivi basé modèle semblable à celui proposé dans [2]. En bas : les résultats obtenus avec le SLAM séquentiel qui utilise l'algorithme de raffinement LBA_CS&E.

forme des poignées, *etc* tout en se déplaçant autour. Chaque partie de la cuisine, examples le revêtement, le mobilier, le plan de travail, peut être modifiée séparément ou simultanément avec d'autres. Lorsque l'utilisateur sélectionne un élément de la cuisine, un menu apparait dans les images. Il contient les différents designs disponibles en magasin pour l'élément en question. Une fois que le design est choisi, la cuisine est automatiquement augmentée en temps réel. Un rendu réaliste est obtenu en modélisant les sources lumineuses de la salle ainsi que la lumière provenant de l'extérieur. Notons qu'il est également possible d'ajouter du mobilier ou éventuellement d'en remplacer. La Figure 8 représente les différentes fonctionnalités du logiciel de design de cuisine.

7 Conclusion

Ce papier présente un processus d'ajustement de faisceaux contraint par segments pour les algorithmes SLAM basé images clefs. Une fonction de coût composée prenant en compte les informations issues du modèle géométrique et les relations multi-vues des parties connue et inconnue de l'environnement, a été proposée.
Des résultats expérimentaux, sur des données de synthèse, montrent que l'approche proposée fournit de meilleurs résultats que le SLAM "classique", en terme de précision et de robustesse à une initialisation approximative.
Une comparaison est ensuite effectuée, sur des données réelles entre l'approche proposée SLAM avec LBA_CS&E et l'état de l'art en suivi d'objet 3D. Cette comparaison montre que l'approche proposée fournit de meilleurs résultats que les algorithmes de suivi basés modèle en terme de stabilité. En effet, la prise en compte de l'environnement permet de stabiliser la localisation et de conserver le suivi,

FIGURE 7 – Réalité augmentée sur une voiture miniature non texturée par le SLAM avec l'algorithme de raffinement LBA_CS&E. De gauche à droite : la voiture avec sa couleur originale et la même augmentée avec différentes couleurs de carrosserie. Notons que seule la carrosserie a été changée, les roues, les fenêtres et le pare-brise sont réels.

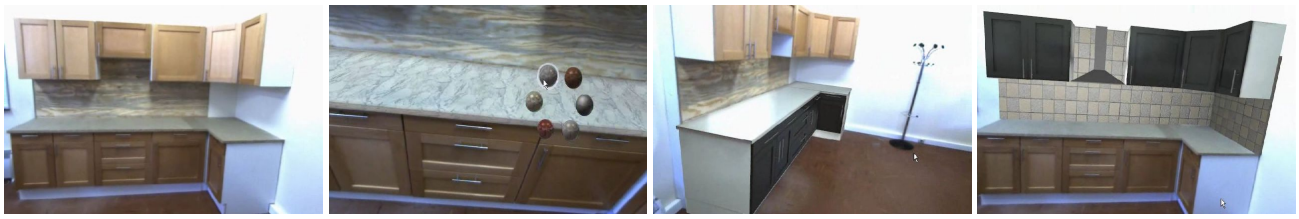

FIGURE 8 – Application de design de cuisine par réalité augmentée. La cuisine originale (à gauche) est modifiée, en ligne, de manière virtuelle avec un autre plan de travail, d'autres couleurs pour le mobilier et avec l'ajout ou le remplacement de mobilier de manière virtuelle. Un menu est proposé pour chaque composant de la cuisine afin de choisir le design à lui appliquer.

même si l'objet d'intérêt est peu ou pas visible.
Pour les travaux à venir nous étudions la possibilité d'intégrer les segments du modèle 3D dans les différentes étapes du SLAM telle que le calcul de pose à chaque image.

Références

[1] Gabriele Bleser, Harald Wuest, and Didier Stricker. Online camera pose estimation in partially known and dynamic scenes. In *ISMAR*, 2006.

[2] Tom Drummond and Roberto Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(7) :932–946, 2002.

[3] C. Harris. Tracking with rigid objects. In *MIT Press*, 1992.

[4] Richard I. Hartley and Peter F. Sturm. Triangulation. In *CAIP*, 1995.

[5] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, 2007.

[6] Georg Klein and David Murray. Improving the agility of keyframe-based slam. In *ECCV*, 2008.

[7] Vincent Lepetit and Pascal Fua. Monocular model-based 3d tracking of rigid objects : A survey. In *FTCGV*, 2005.

[8] Manolis I. A. Lourakis and Antonis A. Argyros. Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment ? In *ICCV*, 2005.

[9] D. Marquardt. An algorithm for least-squares estimation of non linear parameters. *J. Soc. Industr. Appl. Math.*, 11(1) :431–444, 1963.

[10] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3d reconstruction. In *CVPR*, 2006.

[11] Eric Royer, Maxime Lhuillier, Michel Dhome, and Thierry Chateau. Localization in urban environments : Monocular vision compared to a differential gps sensor. In *CVPR*, 2005.

[12] Mohamed Tamaazousti, Vincent Gay-Bellile, Sylvie Naudet-Collette, Steve Bourgeois, and Michel Dhome. Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. In *CVPR*, pages 3073–3080, 2011.

[13] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *ICCVW : International Workshop on Vision Algorithms Theory and Practice*, 2000.

[14] Luca Vacchetti, Vincent Lepetit, and Pascal Fua. Combining edge and texture information for real-time accurate 3d camera tracking. In *ISMAR*, 2004.

[15] Harald Wuest and Didier Stricker. Tracking of industrial objects by using cad models. *Journal of Virtual Reality and Broadcasting*, 4(1), 2007.

[16] Harald Wuest, Florent Vial, and Didier Stricker. Adaptive line tracking with multiple hypotheses for augmented reality. In *ISMAR*, 2005.